Computational Power of a Memory-Based Paralled Computation Model with Communication

AUTHOR(S):

Takenaga, Yasuhiko; Yajima, Shuzo

数理解析研究所講究録
第796巻 1992年 31-42

Computational Power of a Memory-Based Parallel Computation Model with Communication

Yasuhiko Takenaga and Shuzo Yajima

武永康彦 矢島脩三

Departmemt of Information Science, Faculty of Engineering, Kyoto University

1 Introduction

Although memories are originally for storing data, several computations can be performed by adding some functions to memories. Such memories are called functional memories. Because memories can be highly integrated, extremely parallel computation can be realized using the functional memories.

We have proposed memory-based parallel computation models, which consist of a random access machine (RAM) and a new functional memory [1, 2]. The functional memory can access multiple words in parallel according to the partial match with their addresses. The functional memory is used as a SIMD (Single Instruction Multiple Data stream) type parallel computation engine. An FRAM (RAM with Functional memory) which has a simple functional memory is a basic model of memory-based parallel computation. Each word of the functional memory consists of an index field that stores its memory address and a 1-bit data field. We showed that the class of sets accepted by the FRAM within polynomial time is identical with Δ_2^P, that is, the class of sets accepted by a polynomial time bounded deterministic oracle Turing machine with an NP-oracle [3].

On many of the parallel computation models, PSPACE is accelerated to polynomial time. Vector machines [4, 5] are SIMD type computation models of the power. Compared with the models, the communication on the FRAM is limited so that no word can know the contents

of other words. It seems to be the factor to bound the computational power of the FRAM.

In this paper, we consider communicate functions on the functional memory and investigate the effect to the computational power. The cube-FRAM model, which we propose in this paper, is an extended model of the FRAM and has a hypercube network on the functional memory. It has been studied on the computation using hypercube networks. However, on the cube-FRAM model, each word of the functional memory can store only one data bit, and so the operations on the functional memory are very primitive and cannot execute even basic instructions of a RAM.

We show in this paper that the class of sets accepted by the cube-FRAM within polynomial time is identical with PSPACE. Also an extended model of the cube-FRAM, called cube-CAFRAM (RAM with Content Addressable Functional memory) which has infinite data bits in each word, has the same computational power. The results reveal that the even simple processing elements like a word of the functional memory can achieve the same computational power as RAM's on a hypercube network. We can regard that the operations on the functional memory are, in a sense, the essential ones for SIMD type parallel computation to realize the computational power.

Researches on functional memories have been made since more than 30 years ago [6]. Content addressable memory (CAM) is a kind of functional memory. Highly integrated CAMLSI's are developed in recent years [7, 8]. High speed processing using CAM's attracts interests of many researchers [9, 10, 11, 12]. An efficient algorithm for unification using a CAM is proposed in [10]. In this algorithm, memory addresses are stored in a CAM as initial data. Yasuura et al. proposed a functional memory type parallel processor architecture [11]. In addition to the functions of a CAM, the functional memory of [11] has a function of masked search for addresses. We think the researches on memory-based parallel computation models give a theoretical basis for the computation using the functional memory.

The functional memory of the FRAM is realizable by some modification of a random access memory or a CAM [13]. Using today's technology, it seems possible to realize a functional memory LSI with the capacity of several megabits. It may be hard to construct a large hypercube network on the functional memory. However, in practice, more realizable networks like a binary tree work as powerful as a hypercube network, and even a collection of small networks can accelerate computations of many problems.

In Section 2, we present the basic model FRAM. In Section 3, the cube-FRAM model which has a hypercube network on the functional memory is proposed, and its computational power is considered in Section 4.

2 FRAM : The Basic Model of Memory-Based Parallel Computation

The FRAM is the most basic memory-based parallel computation model, which has only limited instructions on the functional memory. On the functional memory of the FRAM, parallel access to multiple data is possible according to the partial match with their memory addresses.

The FRAM model consists of a RAM, an unbounded functional memory and a search result register, as shown in Fig.1.

As shown in Fig.2, the functional memory consists of unbounded number of words, and each of them consists of an index field and a 1-bit data field. In the index field, binary numbers 0,1,2,... are stored in order from the first word. The number in the index field is called an address. The addresses cannot be rewritten. With a search argument and a mask data, a parallel masked search can be performed for the index fields of all the words. The bits of the index field where the mask data have 1 are masked. The contents of the data fields can be

rewritten. In Fig.2, two words which are framed with wide lines are matched with the search argument.

The search result register is a 1-bit flag. 0 or 1 is loaded automatically into the register according to the result of a search instruction.

The instructions of the FRAM is the same as those of an ordinary RAM except SEARCH0 and WRITE1 instructions. The search result register can be an operand of the instructions. The program is not stored in the registers and cannot be rewritten.

SEARCH0 and WRITE1 are the instructions that use the functional memory. The first operand is a search argument and the second operand is a mask data. The contents of the operands are regarded as binary numbers and 0's are assumed in higher bits out of consideration. Namely, when n-bit binary numbers are given as operands, the 2^n memory cells with addresses from 0 through $2^n - 1$ are processed. A SEARCH0 instruction performs a parallel masked search and inspects if there exists a word whose data field stores 0 among the matched words. If it exists, 1, otherwise 0, is set in the search result register. This operation does not change the contents of the data fields. A WRITE1 instruction writes 1 into the data fields of all the matched words. The contents of the data fields are not changed in the unmatched words.

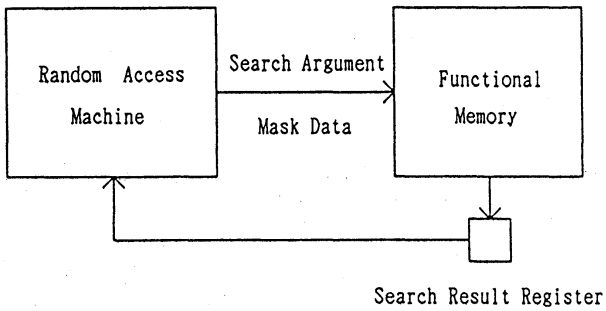

Fig. 1 The scheme of the FRAM.

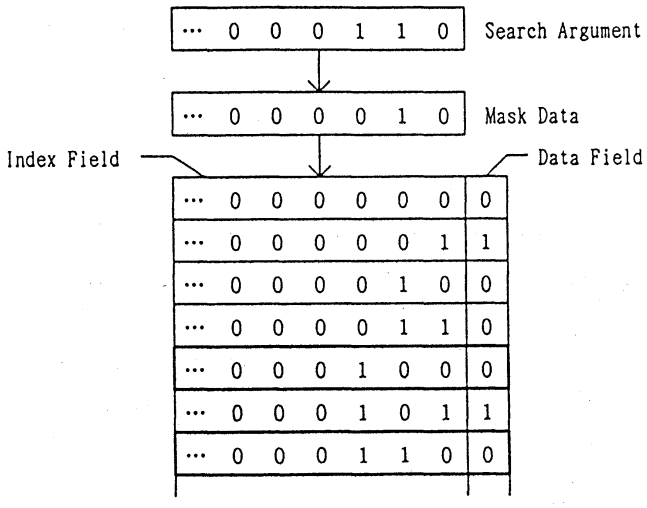

Fig. 2 Searching on the functional memory.

The computation time on the FRAM is defined to be the number of executed instructions as ordinary RAM's.

The FRAM model realizes a completely SIMD type parallel computation. Each word of the functional memory is a local processing element and controlled by the random access machine. We want to emphasize that the instructions of the FRAM are very simple. Both the function of local computation and the way of communication between them, realized by WRITE1 and SEARCH0, are quite restricted.

We have shown the following theorem on the computational power of the FRAM [1].

Theorem1 The class of sets accepted by the FRAM within polynomial time is identical with Δ_2^P.

3 Model with Communication

3.1 Communication on the Functional Memory

On the FRAM, the way of communication is restricted, as compared with bus connected processor models, so that no word can receive the communicated value. Therefore, if a word requires to refer the contents of another word, it is necessary to read it out to the random access machine. Thus, it is impossible to communicate between the words in parallel for the words. It is supposed to be the factor to limit the computational power of the FRAM.

In the CAM of [7], communication between the adjacent words is realized. The results of searching can be shifted in parallel to the next words. It is used to deal with multiple words like a single word and manage large data over the size of a word.

If the communication of this kind is added to the FRAM (or the CAFRAM), it accelerates the computation of many practical problems [14]. However, in the polynomial time computation, acceptable class does not increase because each word can refer the contents of k words

by k times of communication. In this paper, we think of a more powerful communication on the hypercube network.

3.2 Cube-FRAM Model

In this section, we propose a cube-FRAM model. The cube-FRAM has instructions to communicate between the words in parallel, in addition to the operations on the FRAM. In the following, we define the communicate functions of the cube-FRAM.

The lines to connect the words construct a hypercube network. Each word is connected with the words whose addresses are different from its address at only one bit. For example, the word with address 0 is connected with the words with addresses 1,2,4,8, etc.

The cube-FRAM has communicate instructions COMOR and COMAND. In the communicate instructions, a bit position of the address is appointed as an operand. The communication is executed in parallel between all the pairs of words whose addresses differ only at the bit position. For example, when the lowest bit is appointed in a communicate instruction, communications are executed between the words 0 and 1, 2 and 3, 4 and 5, etc. In the COMAND instruction, logical product of the data bits is computed and stored into their data fields. In the COMOR instruction, logical sum is computed and stored into their data fields. These instructions are executed in a unit time.

The way of communication on the cube-FRAM preserves the SIMD type parallelism of the FRAM.

4 Computational Power of the cube-FRAM

4.1 Algorithm for QBF

The QBF problem is one of the representative PSPACE-complete problems [15].

QBF : Is a quantified Boolean formula true?

A quantified Boolean formula is

$Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \ F(x_1, x_2, \cdots, x_n)$,

where $F(x_1, x_2, \cdots, x_n)$ is a Boolean formula with variables $x_1, x_2, \cdots, x_n$ and $Q_m \in \{\exists, \forall\}$ $(1 \leq m \leq n)$. □

In this section, we show a polynomial time algorithm for QBF on the cube-FRAM.

In the following algorithm, we assume that the Boolean formula is in CNF. QBF is still a PSPACE-complete problem under the constraint. Here, we consider k-variable QBF.

[Algorithm QBF]

Step1 : Perform following (1) and (2) for each sum-term.

(1) Make a search argument and a mask data according to each sum-term. In the mask data, let the bits corresponding to the variables included in the sum-term be 0 and the others be 1. In the search argument, let the bits corresponding to the variables appearing as negative literals in the sum-term be 1 and the others be 0.

(2) Execute a WRITE1 instruction with the search argument and the mask data made in (1).

Step2 : Repeat the following operation for m=k to 1.

Execute a communicate instruction appointing the bit which corresponds to x_m. The instruction is COMOR if the quantifier of x_m is $\forall$, and COMAND if it is $\exists$.

Step3 : (1) Make a mask data such that the lowest k bits are 1 and the others are 0.

(2) Execute a SEARCH0 instruction with the search argument made in (1) and 0 as a mask data.

(3) If the content of the search result register is 1, the answer is 'yes'. □

In Step 1, for each given sum-term, we check the assignments for variables which do not satisfy the sum-term by writing 1 in the corresponding words. For example, for a sum-term $x_5 + \bar{x_2} + x_1$ in a five variable CNF Boolean formula, we make the mask data 0...01100 and the search argument 0...00010. After Step 1, the data bit is 0 when the assignment satisfies the Boolean formula, and otherwise 1. After performing Step2 until $m = i$, the content of a data bit is determined only by the address bits corresponding to $x_1, \cdots, x_{i-2}$ and x_{i-1}. The value is 0 when $Q_i x_i \cdots Q_n x_n \ F(x_1, x_2, \cdots, x_n)$ is true. After Step2, all of the 2^k data bits store the same value. If the quantified Boolean formula is true, the value is 0.

Step1 requires linear time. Step2 and Step3 takes time proportional to k. Then [Algorithm QBF] is a linear time algorithm.

4.2 Computational Power of the cube-FRAM

We will show the following theorem on the computational power of the cube-FRAM.

Theorem2 The class of sets accepted by the polynomial time bounded cube-FRAM is identical with PSPACE.

Proof A PSPACE-complete problem QBF can be solved within polynomial time on the cube-FRAM. Thus we can think that the cube-FRAM can work as a random access machine with a PSPACE-oracle. It is clear that the class of sets accepted by the polynomial time bounded cube-FRAM includes PSPACE. The converse is shown as follows.

We simulate a polynomial time bounded cube-FRAM by a random access machine. The

operations of the cube-FRAM which do not use the functional memory are easily simulated within polynomial space. The results of the operations on the functional memory can be referred only through the search result register. Thus, we have to compute the contents of the data fields only when a SEARCH0 instruction is executed. Compute the data bit and examine if it is 0 for each word in turn. The value of a data bit in word A is computed by the following recursive algorithm. Let the word with which A communicated at the last communicate instruction be word B.

Step1: Compute the value of the data bit in word A before the last communicate instruction.

Step2: Compute the value of the data bit in word B before the last communicate instruction.

Step3: Compute the present value of the data bit in word A using the result of Step1 and Step2.

The necessary space for this algorithm is proportional to the product of the number of used address bits and the number of executed communicate instructions. Therefore, a polynomial time bounded cube-FRAM is simulated within polynomial space. □

This speed up is obtained because each word can refer the contents of 2^k words by k times of communicate instructions. If a network satisfies the condition, it may be used instead of the hypercube network [16]. For example, we can easily see that a binary tree network can take the place of a hypercube network, because the algorithm for QBF requires communications only on the binary tree structure. The number of connections is $O(n)$ on the binary tree network, and that on the hypercube network is $O(nlogn)$, where n is the number of nodes. Therefore, the binary tree network is more effective and easy to realize. However, it may disturb the SIMD type parallelism of the memory-based parallel computation models.

4.3 Cube-CAFRAM : An Extended Model

The cube-FRAM has minimum communicate functions to accelerate PSPACE to polynomial time. Here, we consider a cube-CAFRAM model which has functions like CAM's and a communicate function.

The cube-CAFRAM is based on the CAFRAM model [2]. Each word of its functional memory has infinite data bits and parallel masked search is performed both in the index field and the data field. In a SEARCH instruction, it is found if there is a matched word or not. In a WRITEBIT instruction, it rewrites the data bits according to the result of searching.

COM instruction is added to the instruction set of the CAFRAM, which is a communicate instruction on the hypercube network. The lowest bit of the data field is exchanged between the pairwised words by the instruction.

The class of sets accepted by the polynomial time bounded cube-CAFRAM is identical with PSPACE. The proof is almost the same as the case of the cube-FRAM. This model is realistic and convenient to make algorithms on it. We can observe that satisfiability of any Boolean formula with k-variables is computed on the CAFRAM in linear time using 2^k words. Therefore, QBF with any Boolean formula is solved in linear time on the cube-CAFRAM.

5 Conclusion

In this paper, we have proposed the memory-based parallel computation model with communication called a cube-FRAM, and considered their computational power. In addition to the parallel masked search for their memory addresses, the cube-FRAM model can communicate between the words of the functional memory on the hypercube network. We showed that PSPACE is accelerated to polynomial time on the model. The operations on the functional memory of the cube-FRAM are very simple and, in a sense, essential ones to realize the

computational power on a SIMD type parallel computation model.

References

[1] N. Takagi, Y. Takenaga and S. Yajima : "A Memory-Type Parallel Computation Model and Its Computational Power" Trans. IPSJ, 31, 11, pp.1565-1571(1990).

[2] Y. Takenaga, N. Takagi and S. Yajima : "A Memory-Based Parallel Computation Model with a Content Addressable Memory and Its Computational Power," Report of Technical Group on Theoretical Foundations of Computing, IEICE, COMP89-118 pp.39-44(1988).

[3] L. J. Stockmeyer : "The Polynomial Time Hierarchy," Theoretical Computer Science, 3, 1, pp.1-22(1977).

[4] V. R. Pratt, M. O. Rabin and L. J. Stockmeyer : "A Characterization of the Power of Vector Machines," 6th STOC, pp.122-134(1974).

[5] K. Iwama : "A Canonical Form of Vector Machines," Report of Technical Group on Theoretical Foundations of Computing, IEICE, COMP88-22, pp.53-58(1988).

[6] T. Kohonen : Content Addressable Memories 2nd ed., Springer-Verlag (1987).

[7] T. Ogura, J. Yamada, S. Yamada and M. Tan-no : "A 20-kbit Associative Memory LSI for Artificial Intelligence Machines," IEEE J. Solid-State Circuits, vol.24, no.4, pp.1014-1020(1989).

[8] T. Hamamoto, T. Yamagata, M. Mihara, T. Kobayashi and M. Yamada : "A 288kbit Fully Parallel Content Addressable Memory Using Stacked Capacitor Cell Structure," IEICE National Convention Record, Spring'91, C-649 (1991).

[9] L. Chisvin and R. J. Duckworth, : "Content-addressable and Associative Memory : Alternatives to the Ubiquitous RAM," IEEE Computer, 22, pp.51-64(1989).

[10] M. Ohkubo, H. Yasuura, N. Takagi and S. Yajima : "A Hardware-Oriented Unification Algorithm Using a Content Addressable Memory," Trans. IPSJ, 28, 9, pp.915-922(1987).

[11] H. Yasuura, T. Tsujimoto and K. Tamaru : "Functional Memory Type Parallel Processor Architecture for Combinational Problems," Trans. IEICE, vol.J72-A pp.222-230(1989).

[12] A. Kokubu, T. Higuchi and T. Furuya : "Parallel Associative Memory System of Semantic Network Machine," Report of Technical Group on Architecture, IPSJ, ARC80-9, pp.65-72(1990).

[13] N. Takagi, Y. Takenaga and S. Yajima : "On a New Supercomputer with a Memory-Based Parallel Computation Engine," Report of Technical Group on Architecture, IPSJ, ARC80-13 pp.97-103(1990).

[14] N. Ishiura and S. Yajima : "Linear Time Fault Simulation Using a Content Addressable Memory," Proc. of the 4th KARUIZAWA Workshop on Circuits and Systems, pp.63-68(1991).

[15] J. E. Hopcroft, and J. Wyllie : Introduction to Automata Theory, Languages and Computation, Addison-Wesley R.M. (1979).

[16] Y. Takenaga and S. Yajima : "The Topology of Networks on a Memory-Based Parallel Computation Model," Report of Technical Group on Theoretical Foundations of Computing, IEICE, COMP90-71, pp.53-58(1988).